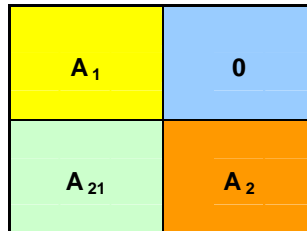


Block-Triangular Matrix Form

Square sparse matrices, with same zero elements, can be, under certain conditions, put in the useful form called “block-triangular” (also called “Jordan’s form”) by simple permutations of rows and columns

1	2	1	0	0	0
2	1	5	0	0	0
1	-1	3	0	0	0
-6	5	3	1	1	2
1	-3	2	1	-1	-2
-9	7	1	1	2	1



The block-triangular form allows a good saving of computation effort for many important problems of linear algebra: linear system solving, determinant computing, eigenvalues problems, etc. We have to point out that each of these tasks has a computing cost that grows approximately with N^3 . Thus, reducing for example the dimension to $N/2$, the effort will decrease 8 times. Clearly it’s a great advantage.

Linear system solving

For example, the following (6 x 6) linear system

$$A x = b$$

1	2	1	0	0	0	x_1	b_1
2	1	5	0	0	0	x_2	b_2
1	-1	3	0	0	0	x_3	b_3
-6	5	3	1	1	2	x_4	b_4
1	-3	2	1	-1	-2	x_5	b_5
-9	7	1	1	2	1	x_6	b_6

It could be written as

$$A_1 x_1 = b_1$$

$$A_2 x_2 = b_2 - c_2$$

where the vector c_2 is given by: $c_2 = A_{21} x_1$

Practically, the original system (6 x 6) is split into two (3 x 3) sub-systems

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 5 \\ 1 & -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 2 \\ 1 & -1 & -2 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_4 \\ b_5 \\ b_6 \end{bmatrix} - \begin{bmatrix} -6 & 5 & 3 \\ 1 & -3 & 2 \\ -9 & 7 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Determinant computing

Determinant computing also takes advantage from the block-triangular form
 For example, the determinant of the following (6 x 6) is given by the determinants product of the two matrices (3 x 3) A_1 and A_2 .

$$\begin{bmatrix} 1 & 2 & 1 & 0 & 0 & 0 \\ 2 & 1 & 5 & 0 & 0 & 0 \\ 1 & -1 & 3 & 0 & 0 & 0 \\ -6 & 5 & 3 & 1 & 1 & 2 \\ 1 & -3 & 2 & 1 & -1 & -2 \\ -9 & 7 & 1 & 1 & 2 & 1 \end{bmatrix} = 18$$

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 5 \\ 1 & -1 & 3 \end{bmatrix} = 3$$

$$\begin{bmatrix} 1 & 1 & 2 \\ 1 & -1 & -2 \\ 1 & 2 & 1 \end{bmatrix} = 6$$

Permutations

Differently from the other factorization algorithms (Gauss, LR, etc.) the block-triangular reduction use only permutations of rows and columns. From the point of view of the linear algebra a permutation can be treated as a similary transformation.
 For example, given a (6 x 6) matrix, the exchange between the rows 2 and 5, followed by the exchange between the columns 2 and 5, can be formally (but only formally) written as.

$$\mathbf{B} = \mathbf{P}^T \mathbf{A} \mathbf{P} \quad , \quad \text{where the permutation matrix is } \mathbf{P} = (e_1, e_5, e_3, e_4, e_2, e_6)$$

A					
1	0	0	1	2	0
1	-1	1	2	-3	-2
-6	1	1	3	5	2
1	0	0	3	-1	0
2	0	0	5	1	0
-9	2	1	1	7	1

P					
1	0	0	0	0	0
0	0	0	0	1	0
0	0	1	0	0	0
0	0	0	1	0	0
0	1	0	0	0	0
0	0	0	0	0	1

$P^T A P$					
1	2	0	1	0	0
2	1	0	5	0	0
-6	5	1	3	1	2
1	-1	0	3	0	0
1	-3	1	2	-1	-2
-9	7	1	1	2	1

Remark. From the point of view of the numeric calculus the matrix multiplication is a very expensive task that we should be avoided; we use instead the direct exchange of the rows and columns or, even better, the exchange of the indices.

We have to point out that similarity transform keeps the original eigenvalues. Consequently the eigenvalues of the matrix **A** are the same of the matrix **B**

Eigenvalues Problem

The eigenvalue problem also takes advantage from the block-triangular form.

For example, the following matrix (6 x 6) **A** has the eigenvalues:

$$\lambda = [-7, -1, 1, 2, 3, 5]$$

A	λ	A₁	A₂	λ_1	λ_2																																																																		
<table style="width: 100%; border-collapse: collapse;"> <tr><td>-15</td><td>0</td><td>-16</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>10</td><td>2</td><td>11</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>8</td><td>0</td><td>9</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>3</td><td>5</td><td>3</td><td>0</td><td>-4</td></tr> <tr><td>2</td><td>6</td><td>1</td><td>2</td><td>5</td><td>4</td></tr> <tr><td>-4</td><td>9</td><td>-3</td><td>-6</td><td>-6</td><td>-1</td></tr> </table>	-15	0	-16	0	0	0	10	2	11	0	0	0	8	0	9	0	0	0	1	3	5	3	0	-4	2	6	1	2	5	4	-4	9	-3	-6	-6	-1	<table style="width: 100%; border-collapse: collapse;"> <tr><td>-7</td></tr> <tr><td>-1</td></tr> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>3</td></tr> <tr><td>5</td></tr> </table>	-7	-1	1	2	3	5	<table style="width: 100%; border-collapse: collapse;"> <tr><td>-15</td><td>0</td><td>-16</td></tr> <tr><td>10</td><td>2</td><td>11</td></tr> <tr><td>8</td><td>0</td><td>9</td></tr> </table>	-15	0	-16	10	2	11	8	0	9	<table style="width: 100%; border-collapse: collapse;"> <tr><td>3</td><td>0</td><td>-4</td></tr> <tr><td>2</td><td>5</td><td>4</td></tr> <tr><td>-6</td><td>-6</td><td>-1</td></tr> </table>	3	0	-4	2	5	4	-6	-6	-1	<table style="width: 100%; border-collapse: collapse;"> <tr><td>1</td></tr> <tr><td>2</td></tr> <tr><td>-7</td></tr> </table>	1	2	-7	<table style="width: 100%; border-collapse: collapse;"> <tr><td>-1</td></tr> <tr><td>3</td></tr> <tr><td>5</td></tr> </table>	-1	3	5
-15	0	-16	0	0	0																																																																		
10	2	11	0	0	0																																																																		
8	0	9	0	0	0																																																																		
1	3	5	3	0	-4																																																																		
2	6	1	2	5	4																																																																		
-4	9	-3	-6	-6	-1																																																																		
-7																																																																							
-1																																																																							
1																																																																							
2																																																																							
3																																																																							
5																																																																							
-15	0	-16																																																																					
10	2	11																																																																					
8	0	9																																																																					
3	0	-4																																																																					
2	5	4																																																																					
-6	-6	-1																																																																					
1																																																																							
2																																																																							
-7																																																																							
-1																																																																							
3																																																																							
5																																																																							

The eigenvalues set of the (6 x 6) matrix **A** is the sum of the eigenvalues set of **A₁** [1 , 2 , -7] and the eigenvalues set of **A₂** [-1 , 3 , 5].

Several kinds of block-triangular form

Up to now the matrices that we have seen are only one kind of block-triangular; but there are many other block-triangular schemas having blocks with different dimension each others. At last, all the blocks can have unitary dimension as in lower-triangular matrix.

Just below there are some example of block-triangular matrices (blocks are yellow)

<table style="width: 100%; border-collapse: collapse;"> <tr><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> </table>	x	x	0	0	0	0	x	x	0	0	0	0	x	x	x	0	0	0	x	x	x	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	<table style="width: 100%; border-collapse: collapse;"> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> </table>	x	x	x	0	0	0	x	x	x	0	0	0	x	x	x	0	0	0	x	x	x	x	0	0	x	x	x	x	x	0	x	x	x	x	x	x	<table style="width: 100%; border-collapse: collapse;"> <tr><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> </table>	x	0	0	0	0	0	x	x	0	0	0	0	x	x	x	0	0	0	x	x	x	x	0	0	x	x	x	x	x	0	x	x	x	x	x	x
x	x	0	0	0	0																																																																																																									
x	x	0	0	0	0																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	x	0	0																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	x	0	0																																																																																																									
x	x	x	x	x	0																																																																																																									
x	x	x	x	x	x																																																																																																									
x	0	0	0	0	0																																																																																																									
x	x	0	0	0	0																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	x	0	0																																																																																																									
x	x	x	x	x	0																																																																																																									
x	x	x	x	x	x																																																																																																									
<table style="width: 100%; border-collapse: collapse;"> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> </table>	x	x	x	0	0	0	x	x	x	0	0	0	x	x	x	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	<table style="width: 100%; border-collapse: collapse;"> <tr><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> </table>	x	x	0	0	0	0	x	x	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	<table style="width: 100%; border-collapse: collapse;"> <tr><td>x</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> <tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr> </table>	x	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
x	x	x	0	0	0																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	0	0	0																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	0	0	0	0																																																																																																									
x	x	0	0	0	0																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	0	0	0	0	0																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									
x	x	x	x	x	x																																																																																																									

Remark. The effort reduction is high when the dimension of the max block is low. In the first matrix the dimension of the max block is 2; in the second matrix is 3; in the third matrix the dimension of the max block is 1, obtaining the best effort reduction that it is possible.

On the contrary, the two last matrices give an effort reduction quite poor.

Permutation matrices

Is it always possible to transform a square matrix into a block-triangular form? Unfortunately not. The chance for block-triangular reduction depends of course by the zero elements. So only sparse matrices can be block-partitioned. But this is not sufficient. It depends also by the zeros configuration of the matrix.

Two important problems arise:

1. To detect if a matrix can be reduced to a block-triangular form
2. To obtain the permutation matrix **P**

Several methods are developed in the past for solving these problems. One very popular is the Flow-Graph method.

Matrix Flow-Graph

Following this method, we draw the graph of the given matrix following these simple rules:

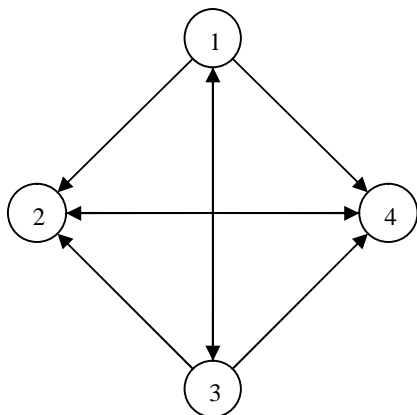
- the graph consists of *nodes* and *branches*
- the number of the nodes is equal to the dimension of the matrix
- the nodes, numbered from 1 to N, represent the elements of the first diagonal a_{ii}
- for each elements $a_{ij} \neq 0$ we draw an oriented branch (arrow) from node-i to node-j

Complicated? Not really. Let's have a look at this example.

Given the matrix **A** (4 x 4):

4	2	3	1
0	-1	0	1
3	1	-1	2
0	1	0	1

The flow-graph $G(\mathbf{A})$ associated, looks like the following :



Where

- The node 1 is linked to the nodes 2, 3, 4.
- The node 2 is linked to the node 4
- The node 3 is linked to the nodes 1, 2, 4
- The node 4 is linked to the node 2

We observe that from the node 2 there is not any path linking the node 1 or the node 3
 Similarly happens if we start from the node 4
 This is sufficient to say that the graph is not strong connected

Flow-Graph rule. If is always possible for each node to find a path going through all other nodes, then we say that the graph is **strong connected**

An important theorem of the Graph Theory states that if the flow-graph $G(\mathbf{A})$ is strong connected then the associate matrix is not reducible to the block-triangular form and vice versa. On the contrary, if the flow-graph $G(\mathbf{A})$ is not strong connected then it always exists a permutation matrix \mathbf{P} that reduces the associate matrix to the block-triangular form. Synthetically:

$$\begin{aligned}
 G(\mathbf{A}) \text{ strong connected} & \Leftrightarrow \text{matrix } \mathbf{A} \text{ irreducible} \\
 G(\mathbf{A}) \text{ not strong connected} & \Leftrightarrow \text{matrix } \mathbf{A} \text{ block reducible}
 \end{aligned}$$

This method is quite elegant and very important in the Graph theory. But from the point of view of the practical calculus it has several drawbacks:

- it becomes laborious for larger matrices
- the software coding is quite complicated
- it does not provide the permutation matrix \mathbf{P}

In the above example, we observe that for $\mathbf{P} = [e_2, e_4, e_1, e_3]$, the similarity transform gives a block-triangular form

$$\mathbf{B} = \mathbf{P}^T \mathbf{A} \mathbf{P}$$

A	P	P^T A P																																																
<table style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px 10px;">4</td><td style="padding: 2px 10px;">2</td><td style="padding: 2px 10px;">3</td><td style="padding: 2px 10px;">1</td></tr> <tr><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">-1</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">1</td></tr> <tr><td style="padding: 2px 10px;">3</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">-1</td><td style="padding: 2px 10px;">2</td></tr> <tr><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">1</td></tr> </table>	4	2	3	1	0	-1	0	1	3	1	-1	2	0	1	0	1	<table style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">0</td></tr> <tr><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td></tr> <tr><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">1</td></tr> <tr><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td></tr> </table>	0	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	<table style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px 10px;">-1</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td></tr> <tr><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">0</td><td style="padding: 2px 10px;">0</td></tr> <tr><td style="padding: 2px 10px;">2</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">4</td><td style="padding: 2px 10px;">3</td></tr> <tr><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">2</td><td style="padding: 2px 10px;">3</td><td style="padding: 2px 10px;">-1</td></tr> </table>	-1	1	0	0	1	1	0	0	2	1	4	3	1	2	3	-1
4	2	3	1																																															
0	-1	0	1																																															
3	1	-1	2																																															
0	1	0	1																																															
0	0	1	0																																															
1	0	0	0																																															
0	0	0	1																																															
0	1	0	0																																															
-1	1	0	0																																															
1	1	0	0																																															
2	1	4	3																																															
1	2	3	-1																																															

For matrices larger than (4 x 4) the effort for searching and testing all possible permutations grows sharply. For example, it requires a heavy work for matrices like the following one.

1	0	0	1	2	0
1	-1	1	2	-3	-2
-6	1	1	3	5	2
1	0	0	3	-1	0
2	0	0	5	1	0
-9	2	1	1	7	1

For this reasons the flow-graph method becomes practically useless for matrices of (7 x 7) or more.

The score-algorithm

(A strange, working, algorithm)

In this chapter we shall introduce a technique for efficiently reducing a sparse matrix to a block-triangular form. The method is both simple and very efficient, and it can be applied also to medium-large matrices. It consists of an iterative process having the main goal to group zeros near the upper-right corner of the matrix using only rows and columns exchanges.

This algorithm was first ideated as automatic program, but thanks to its simplicity it can be also performed by hand, at least, for low-moderate matrices.

Let's see how it works

Giving for example the (6 x 6) matrix just shown above, we begin to initialize the permutation vector

1	2	3	4	5	6
e_1	e_2	e_3	e_4	e_5	e_6

1	0	0	1	2	0
1	-1	1	2	-3	-2
-6	1	1	3	5	2
1	0	0	3	-1	0
2	0	0	5	1	0
-9	2	1	1	7	1

The main goal is to take to the upper triangular area (grey area) the most possible zeros.

Let's begin to search all elements not zero over the first diagonal. The searching must start from the first row and from right to left: thus from the element a_{16} ; if zero, we jump to the near element a_{15} and so on till to a_{12} .

Then we repeat along the second row, from a_{26} to a_{23} . And so on till the last row

	2		5		
1	0	0	1	2	0
1	-1	1	2	-3	-2
-6	1	1	3	5	2
1	0	0	3	-1	0
2	0	0	5	1	0
-9	2	1	1	7	1

In this example, the first element not zero is a_{15} ;

Let's search, if exists, the first zero on the same row, beginning from left to right.

The first 0 is the element a_{12} . We shall exchange the columns 2 e 5 and after, the rows 2 e 5

After the permutation (2, 5), the matrix will be the following:

	1	5	3	4	2	6
A	1	0	0	1	2	0
	1	-1	1	2	-3	-2
	-6	1	1	3	5	2
	1	0	0	3	-1	0
	2	0	0	5	1	0
	-9	2	1	1	7	1
P	1	0	0	0	0	0
	0	0	0	0	1	0
	0	0	1	0	0	0
	0	0	0	1	0	0
	0	1	0	0	0	0
	0	0	0	0	0	1
P^TAP	1	2	0	1	0	0
	2	1	0	5	0	0
	-6	5	1	3	1	2
	1	-1	0	3	0	0
	1	-3	1	2	-1	-2
	-9	7	1	1	2	1

We observe the zero grouping close to the upper-right corner.

		3	4		
1	2	0	1	0	0
2	1	0	5	0	0
-6	5	1	3	1	2
1	-1	0	3	0	0
1	-3	1	2	-1	-2
-9	7	1	1	2	1

Now the first non-zero element starting from right is

a_{14} . The first 0, starting from left, is a_{13} . Thus we permute 3 e 4

After permutation 3, 4 we have:

	1	2	4	3	5	6
A	1	2	0	1	0	0
	2	1	0	5	0	0
	-6	5	1	3	1	2
	1	-1	0	3	0	0
	1	-3	1	2	-1	-2
	-9	7	1	1	2	1
P	1	0	0	0	0	0
	0	1	0	0	0	0
	0	0	0	1	0	0
	0	0	1	0	0	0
	0	0	0	0	1	0
	0	0	0	0	0	1
P^TAP	1	2	1	0	0	0
	2	1	5	0	0	0
	1	-1	3	0	0	0
	-6	5	3	1	1	2
	1	-3	2	1	-1	-2
	-9	7	1	1	2	1

All zeros are now positioned in the upper-triangular area. The matrix is partitioned in two (3 x 3) blocks. The process ends.

The finally permutation matrix is

1	2	3	4	5	6
e1	e5	e4	e3	e2	e6

As shown, with only 2 permutations we were able to reduce in block-triangular form a (6 x 6) matrix. We have to put in evidence that we are worked only by hand. This method keeps a good efficiency also with larger matrices.

Let's have a look to another example.
Reduce, if possible, the following (6 x 6) matrix

↓ (

3	1	-1	1	-5	2
0	-1	0	1	0	0
5	1	1	2	-3	4
0	0	0	1	0	0
1	1	7	-9	13	1
0	1	0	-6	0	1

The first element (0, from right, is: a16
The first element (0, from left, is: a21.
So the pivot columns are 1 and 6

↓ ↓

1	1	0	-6	0	0
0	-1	0	1	0	0
4	1	1	2	-3	5
0	0	0	1	0	0
1	1	7	-9	13	1
2	1	-1	1	-5	3

The first element ≠ 0, from right, is: a14
The first element = 0, from left, is: a13.
So the pivot columns are 3 and 4

↓ ↓

1	1	-6	0	0	0
0	-1	1	0	0	0
0	0	1	0	0	0
4	1	2	1	-3	5
1	1	-9	7	13	1
2	1	1	-1	-5	3

The first element ≠ 0, from right, is: a13
The first element = 0, from left, is: a21.
So the pivot columns are 1 e 3.

Finally we get the block-triangular matrix.

1	0	0	0	0	0
1	-1	0	0	0	0
-6	1	1	0	0	0
2	1	4	1	-3	5
-9	1	1	7	13	1
1	1	2	-1	-5	3

The matrix has been block-partitioned:
There are 3 blocks (1 x 1) and one block (3 x 3)

We observe that this algorithm does not provide any information about the success of the process. It simply stops itself when there are no more elements to permute. At the end of the process, if the result matrix is in block-triangular form, then the original matrix is reducible. Otherwise, it means that the original matrix is irreducible and its flow-graph is strong connected.

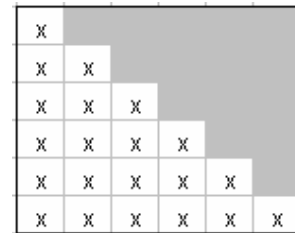
The Score-Function

The matrices used up to now had all zero elements completely filled into the upper-triangle area. Now let's see what happens if the matrix has more zeros than those tightly necessary for block partitioning (spurious zeros). In that case not all permutations will be useful for grouping zeros. Some of them will be useless and some others even will go zeros away from the upper-right corner. Thus, it is necessary to measure the goodness of each permutation. By a simple inspection it is easy to select the "good" permutations from "bad" permutations. But in automatic process it is necessary to choose a function for evaluating the permutation goodness: the *score-function* is the measure adopted in this algorithm.

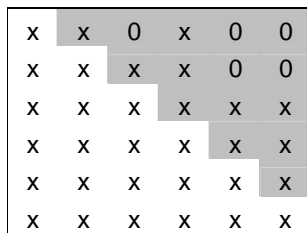
The score function counts the zeros in the upper triangle area (grey) before (A) and after the permutation (B) returning the difference.

$$score = \sum_B w(i, j) - \sum_A w(i, j)$$

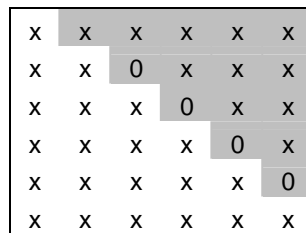
The score will be positive if the permutation will be advantageous otherwise will be negative or null.



The zeros have not the same weight: the zeros nearest to the upper-right corner have a higher weight, because the matrix filled with zeros close to the upper-right corner is better than the one with zeros close to the first diagonal.



better



worse

Apart this concept, the weigh function $w(i,j)$ is arbitrary. One function that we have tested with good result is the following

$$w(i, j) = \begin{cases} 0 & \Leftrightarrow a_{ij} \neq 0 \\ (n - i + 1)^2 \cdot j^2 & \Leftrightarrow a_{ij} = 0 \end{cases} \quad \text{Weight function for a matrix (n x n)}$$

For each permutation recognized, the algorithm measures the score; if positive the permutation is performed, otherwise the permutation is rejected and the algorithm continue to find a new permutation. After some loops the zeros disposition will reach the maximum score possible; every other attempt of permutation will produce a negative or null score. So the algorithm will stop the process.

Example

Now let's see the algorithm in practical cases

1	2	0	2	0	0
0	1	2	0	-3	0
0	0	1	0	5	3
0	3	1	1	0	0
0	0	0	0	1	3
0	0	0	0	1	3

1	3	0	0	0	0
1	3	0	0	0	0
5	3	1	0	0	0
-3	0	2	1	0	0
0	0	1	3	1	0
0	0	0	2	2	1

$P = [e_5, e_6, e_3, e_2, e_4, e_1]$
 Accepted permutations = 6
 Rejected permutations = 4

3	0	0	0	0	0	2	3	0	4
6	1	6	3	0	2	5	1	0	2
0	0	1	0	0	0	1	0	0	0
8	1	8	1	0	0	7	1	0	0
10	1	10	5	0	0	9	1	5	0
0	1	7	4	0	1	6	1	0	3
0	0	2	0	0	0	1	0	0	0
4	0	0	0	0	0	3	1	0	6
9	1	9	4	-1	3	0	1	1	5
5	0	5	0	0	0	0	0	0	1

1	2	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
0	5	1	5	0	0	0	0	0	0
2	0	4	3	3	0	0	0	0	0
3	0	6	4	1	0	0	0	0	0
5	6	2	6	1	1	3	2	0	0
7	8	0	8	1	1	1	0	0	0
6	7	3	0	1	1	4	1	0	0
0	9	5	9	1	1	4	3	1	-1
9	10	0	10	1	1	5	0	5	0

$P = [e_7, e_3, e_{10}, e_1, e_8, e_2, e_4, e_6, e_9, e_5]$
 Accepted permutations = 9
 Rejected permutations = 10

1	0	1	0	1	0	1	6	0	1
1	1	0	1	1	1	1	1	1	0
0	0	1	0	0	0	0	0	0	0
1	0	0	1	1	4	1	1	0	1
0	0	1	0	1	0	5	0	0	0
1	0	1	0	0	1	1	1	0	0
0	0	1	0	0	0	1	0	0	0
0	0	1	0	4	0	1	3	0	0
1	0	1	3	0	4	1	1	1	1
0	0	0	0	0	0	1	4	0	1

1	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
1	5	1	0	0	0	0	0	0	0
1	1	4	3	0	0	0	0	0	0
0	1	0	4	1	0	0	0	0	0
1	1	1	6	1	1	0	0	0	0
1	1	0	1	0	1	1	0	0	0
0	1	1	1	1	1	4	1	0	0
1	1	0	1	1	1	4	3	1	0
0	1	1	1	0	1	1	1	1	1

$P = [e_3, e_7, e_5, e_8, e_{10}, e_1, e_6, e_4, e_9, e_2]$
 Accepted permutations = 7
 Rejected permutations = 1

A

3	0	8	0	0	3	0	3	0	0	0	6	0	0	0	14	8	0	7	0
4	4	0	0	0	6	0	6	0	0	3	9	0	0	0	20	0	0	10	4
0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	2	0
0	0	17	10	10	0	10	0	10	0	0	15	0	10	10	0	17	10	16	0
4	9	16	9	9	11	9	11	9	9	8	14	9	9	9	30	0	9	15	9
0	0	0	0	0	1	0	0	0	0	0	4	0	0	0	10	0	0	20	0
0	0	20	20	0	0	13	20	0	20	12	0	0	13	13	38	20	13	0	13
0	0	0	0	0	2	0	2	0	0	0	20	0	0	0	0	7	0	6	0
4	11	18	0	20	13	11	13	11	11	10	16	11	11	11	34	18	0	17	11
20	5	0	0	0	7	0	0	0	5	0	0	0	0	0	0	1	0	11	5
4	0	9	0	0	0	0	4	0	0	1	0	0	0	0	20	0	0	8	0
0	0	4	0	0	0	0	0	0	0	2	0	0	0	0	4	0	3	0	0
4	6	13	0	0	8	0	8	0	6	5	11	6	0	0	0	0	0	12	6
0	7	14	0	0	9	0	9	0	7	20	12	7	7	0	0	0	0	13	0
4	0	19	12	12	14	12	14	12	0	0	17	0	12	12	36	19	12	18	0
0	0	5	0	0	0	0	0	0	0	0	3	0	0	0	8	5	0	4	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
4	8	15	0	0	10	0	10	0	8	7	13	8	8	0	0	0	8	14	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0
4	0	10	0	0	5	0	5	0	0	2	8	0	0	0	18	0	0	0	3

P^TA P

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	5	3	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	20	0	4	10	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	6	0	20	0	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	8	6	14	3	3	3	0	0	0	0	0	0	0	0	0	0	0	0
0	8	9	0	20	0	4	4	1	0	0	0	0	0	0	0	0	0	0	0
0	0	10	8	18	5	5	4	2	3	0	0	0	0	0	0	0	0	0	0
0	10	0	9	20	6	6	4	3	4	4	0	0	0	0	0	0	0	0	0
1	11	0	0	0	7	0	20	0	5	5	5	0	0	0	0	0	0	0	0
0	12	13	11	0	8	8	4	5	6	6	6	6	0	0	0	0	0	0	0
0	13	14	12	0	9	9	0	20	0	7	7	7	7	0	0	0	0	0	0
0	14	15	13	0	10	10	4	7	8	8	8	8	8	0	0	0	0	0	0
19	18	19	17	36	14	14	4	0	0	0	0	0	12	12	12	12	12	12	12
17	16	17	15	0	0	0	0	0	0	0	0	0	10	10	10	10	10	10	10
20	0	20	0	38	0	20	0	12	13	0	20	0	13	13	13	20	13	0	0
18	17	18	16	34	13	13	4	10	11	11	11	11	11	0	11	0	11	11	20
0	15	16	14	30	11	11	4	8	9	9	9	9	9	9	9	9	9	9	9

P = [e17, e19, e3, e12, e16, e6, e8, e1, e11, e20, e2, e10, e13, e14, e18, e15, e4, e7, e9, e5]
 Accepted permutations = 18
 Rejected permutations = 237

As we can see, also for larger matrices the number of permutations remains quite limited. Regarding this and that the permutation is much faster than any other arithmetic operation in floating point, we can guess the high speed of this algorithm

Leonardo Volpi